Parameter Learning Algorithms for Continuous Model Improvement Using Operational Data

Anders L. Madsen[12], Nicolaj Søndberg Jeppesen[1], Frank Jensen[1], Mohamed S. Sayed[3], Niels Lohse[3], Ulrich Moser[4], Luis Neto[5], and Joao Reis[5]

[1] HUGIN EXPERT A/S, Denmark
[2] Department of Computer Science, Aalborg University, Denmark
[3] Loughborough University, United Kingdom
[4] IEF-Werner GmbH, Germany
[5] Instituto de Sistemas e Robtica - Porto (ISR-P), Portugal

Abstract. In this paper, we consider the application of object-oriented Bayesian networks to failure diagnostics in manufacturing systems and continuous model improvement based on operational data. The analysis is based on an object-oriented Bayesian network developed for failure diagnostics of a one-dimensional pick-and-place industrial robot developed by IEF-Werner GmbH. We consider four learning algorithms (batch Expectation-Maximization (EM), incremental EM, Online EM and fractional updating) for parameter updating in the object-oriented Bayesian network using a real operational dataset. Also, we evaluate the performance of the considered algorithms on a dataset generated from the model to determine which algorithm is best suited for recovering the underlying generating distribution. The object-oriented Bayesian network has been integrated into both the control software of the robot as well as into a software architecture that supports diagnostic and prognostic capabilities of devices in manufacturing systems. We evaluate the time performance of the architecture to determine the feasibility of on-line learning from operational data using each of the four algorithms.

Keywords: Bayesian networks, parameter update, practical application

1 Introduction

The need for diagnostic and health monitoring capabilities in manufacturing systems is becoming increasingly important as manufacturing organisations continuously aim to reduce system downtime and unpredicted disturbances to production. We have found that Bayesian networks (BNs) [17, 3, 5] and their extension Object-Oriented Bayesian Networks (OOBNs) [7, 13] are well-suited to capture and represent uncertainty in root cause analysis using both component-level models and wider system-level models integrating component-level models. The crucial need for diagnostic and health monitoring capabilities is accompanied with the availability of increasing amounts of sensory data and decreasing costs of computation on the shop-floor level have opened new opportunities for component suppliers and system integrators to provide more competitive functionalities that go beyond traditional control and process monitoring capabilities

In this paper, we consider the challenge of parameter learning for continuous model improvement using operational data. In particular, we investigate the use of four different approaches to improve the diagnostic performance of an OOBN using operational data. The four algorithms are the batch EM algorithm, incremental EM, Online EM and fractional updating. The investigation is performed using an OOBN for root cause analysis of a pick-and-place industrial robot developed by IEF-Werner GmbH[6] (the Linear Axis). An initial OOBN for root cause analysis has been developed based on expert knowledge [9]. The OOBN has been integrated into the control software of the component and is being deployed in a production line where efficient and effective root cause analysis is required in case of failure. In order to improve the diagnostic performance of the OOBN different methods for continuous model update based on operational data are being investigated. This paper reports on the results of these investigations.

Inspired by the work of [18], a number of approaches are considered. Notice that our work differs from the work of [18] in three important ways: (1) we are considering parameter learning in OOBNs, (2) the objective is to improve diagnostic performance (not classification), and (3) while [18] compare three algorithms, we investigate four algorithms. We consider the EM algorithm [8] for parameter learning from a batch of data (referred to as batch EM). Using batch EM, the idea is to collect data in batches and learning parameters off-line, for instance, during maintenance hours as suggested by [18]. We use batch EM as a reference. Adaptive causal probabilistic networks and fractional updating are described in [16] who cites [21] while adaptive probabilistic networks are described in [19] and [1]. A simliar gradient descent approach is described in [4]. [12] describes how the approach of [16] referred to as sequential learning has been implemented in the HUGIN tool. The online EM algorithm [2] is a stochastic gradient method that is faster than other gradient methods such as [19] which involves a difficult task of determining the step size between iterations.

2 Preliminaries and Notation

A BN $\mathcal{N} = (\mathcal{X}, G, \mathcal{P})$ consists of a directed, acyclic graph G specifying dependence and independence relations over a set of variables and a set of conditional probability distributions (CPDs) $\mathcal{P}$ encoding the strengths of the dependence relations effectively combining elements of probability and graph theory. A BN is a representation of a joint probability distribution $P(\mathcal{X}) = P(X_1, \ldots, X_n) = \prod_{X_i \in \mathcal{X}} P(X_i|\pi_{X_i})$ where π_X are the parents of X in G. The CPD $P(X|\pi_X)$ consists of one probability distribution over the states of X for each configuration of π_X. We consider only discrete variables, which simplifies the presentation.

An OOBN is a Bayesian network augmented with network classes, class instances and an associated notion of interface and private variables [7, 13, 5]. A

[6] http://www.ief-werner.de

class instance is the instantiation of a network class representing a sub-network within another network class. The variables $\mathcal{X}(C)$ of network class C are divided into disjoint subsets of input $\mathcal{I}$, output $\mathcal{O}$ and hidden/private $\mathcal{H}$ variables such that $\mathcal{X}(C) = \mathcal{I} \cup \mathcal{O} \cup \mathcal{H}$ where the interface variables $\mathcal{I} \cup \mathcal{O}$ are used to link nested class instances, see Figure 1.

Inference in an OOBN is performed by creating a *run-time* instance of the model and doing inference in this model. A run-time instance of an OOBN is created by expanding it into a corresponding flat Bayesian network.

To compare the results of learning using different algorithms, the Hellinger distance is used. The Hellinger distance $D_H(P,Q)$ between two probability distributions P and Q is defined as $D_H(P,Q) = \sqrt{\sum_i \sqrt{p_i} - \sqrt{Q_i}}$ [18] who cites [6]. It is similar to the Kullback-Leibler divergence, but defined for zero probabilities. To compare the results of parameter learning using two different algorithms on the same OOBN, the distance is computed as a sum of Hellinger distances. This is similar to the approach taken by [22] and [18]. For each parent configuration π of each node X in each network class C, $D_H(P_1(X|\pi), P_2(X|\pi))$ is computed where P_1 and P_2 are CPDs produced by the two learning algorithms. The values $D_H(P_1(X|\pi), P_2(X|\pi))$ are summed across parent configurations, nodes and classes. In the weighted Hellinger distance $D_H^w(P_1(X|\pi), P_2(X|\pi))$ the $D_H(P_1(X|\pi), P_2(X|\pi))$ is weighted by $P(\pi)$ in the reference model.

3 The Linear Axis

The Linear Axis as a self-sustainable handling system that is designed to be a high performance machine with a demand to work 24h / day seven days a week. Therefore, there is little or no time for maintenance and repair. This means that there is a need for system condition monitoring to prevent failures and for system failure diagnosis. The Linear Axis diagnosis model considered here is used under the assumption that a problem is observed and we need to identify the five most likely root causes. Figure 1 shows the structure of the top-level class of the Linear Axis OOBN. In the figure, blue nodes denote possible root causes, orange nodes denote problem defining nodes, and green nodes denote possible observations such as sensor readings and operator feedback. The model has 39 variables and 716 parameters (CPD entries) and two instances of the network class for the limit switch. The Linear Axis OOBN has been quantified using subject matter expert knowledge. We referred to this model as the knowledge driven model and its development is described in [9].

The diagnosistic performance of the knowledge driven model has been assessed following the approach of [9]. The basic idea, is to iterate through the root causes where each root causes is instantiated to a failure state and all other root causes are instantiated to non-failure. For each such configuration, values for the observations nodes are generated. The values for the observation nodes are propagated in the model and the probabilities of the root causes recorded. This will demonstrate how well the observations can distinguish the root causes.

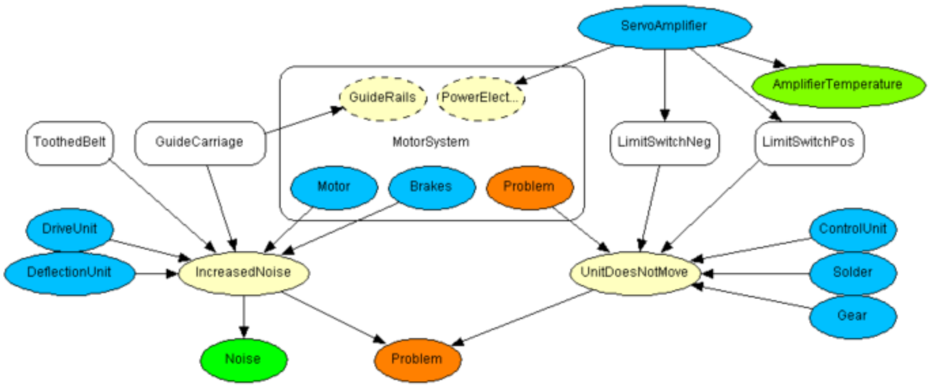


Fig. 1. The top level class of the Linear Axis Model.

4 The SelSus Architecture

The work presented in this paper is part of SelSus European Project, whose objective is the development of a diagnostic and prognosis environment, aware of the condition and history of the machine components within a system or factory for highly effective, self-healing production resources and systems to maximize their performance over longer life times through highly targeted and timely repair, renovation and upgrading. The SelSus architecture defines three levels of abstraction for its constituents: 1) Component Level, which relates directly to machines or its sub-components and is composed of smart sensory capabilities, methods for self-diagnostics and predictive maintenance. 2) Station Level, at this level the developments are constituted by previous capabilities plus human machine interface and tools to support the design and maintenance of the factory station. 3) Factory Level, previous levels capabilities are combined to create a semantic driven maintenance scheduling for large production factory plants.

The Linear Axis, due to its capabilities, is a component that typically integrates a production cell, performing operations in collaboration with other machines (eg: robotic arms and welding tools). To make operational and sensory data available to the SelSus system, the SelComp (SelSus Component) concept was designed. The SelComp (Figure 2), is a self-aware entity that makes available to the SelSus system its internal state conditions, providing this way operational and structural knowledge. A SelComp also provides built-in models for state estimation based on sensor data which enables for pro-active and predictive maintenance. These components have the ability to collect data from sensors that are mounted physically in the same device or in a near location and fuse this data to extend its models capabilities. For this, there are two kinds of SelComps: 1) Machine SelComp [20, 11] (such is the case of Linear Axis, see Figure 3), to the SelSus system it represents a field device, machine or its sub-components. This components use the built-in sensory information, models and algorithms for determining its current condition in terms of failures, diagnostics, prognostics and maintenance. There is also a control layer to prevent a machine from further damage in case of failure, parametrization of operation and to access external sensor data. 2) Sensor SelComp [14, 15], it is designed to provide essentially smart sensor data to SelSus system and more often to Machine SelComp's. This component has *plug&play* capabilities in terms of physical sensors, data models and algorithms.

A crucial part for SelComp to SelComp and SelComp to SelSus System communication is the adopted ontology. The SelSus project define two documents for this purpose *SelSus Self-Description (SSD)* and *Data Payload*. The SSD describes all services and respective variables provided by a SelComp and its specific information (eg: ID, IP, MAC Address, Port information). A service provided by a SelComp is directly related to its internal data processing models and algorithms, the SSD encodes its semantics. A BN for continuous model improvement or failure diagnostics using operational data, can be abstracted by a service to provide outputs to and subscribe inputs from the SelSus System and other SelComp's. The *Data Payload*, identifies the origin of the data (SelComp ID), operation cycle and the service crating data with a specific service ID generated by the SelComp of origin.

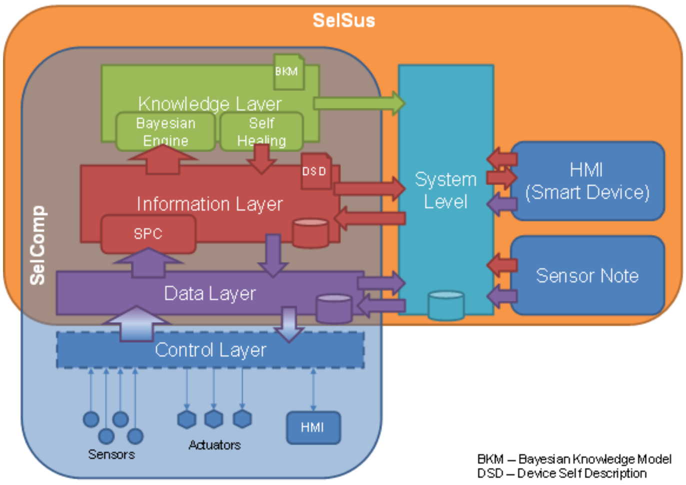


Fig. 2. The SelComp internal architecture concept.

The IEF OOBN model for component-based diagnostic has been encapsulated as a SelComp in the SelSus software architecture 3 to enable system-level diagnostic capabilities [10]. We present the results of a performance evaluation of different levels of integration (direct, local network and wider network).

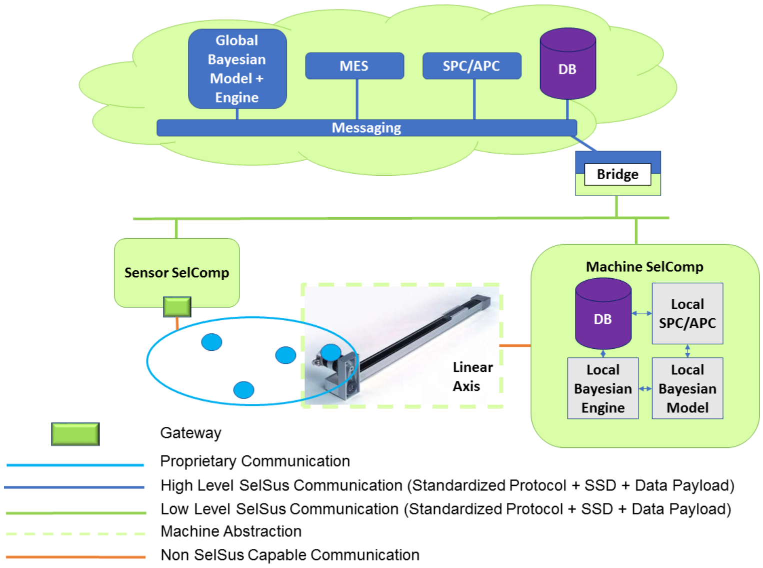


Fig. 3. *SelSus System Architecture.*

5 Parameter Learning Algorithms

here we will describe the four algorithms considered in the work: EM, incremental EM, online EM and fractional updating. The presentation will rather mathematical and we describe the extension of the algorithms to OOBNs.

Data cases $\mathcal{D} = \{c_1, \ldots, c_N\}$ are assumed independent and identically distributed (iid) with values missing at random or completely at random. The underlying model distribution is assumed to be stationary. We let $\theta_{ijk} = P(X_i = k|\pi(X_i) = j)$ or shortly $p(x_{ik}|\pi_{ij})$ denote an entry of a CPD.

The EM algorithm [8] ...

The incremental EM algorithm [18] ...

If cases are complete, then incremental EM and batch EM produces the same results.

The Online EM algorithm [2]

$$m^*_{ijk} = (1 - \gamma)m_{ijk} + \gamma p(x_{ijk}|\epsilon), \tag{1}$$

where m_{ijk} is the normalized sufficient statistics computed as $m_{ijk} = \alpha_{ij} * p(x_{ik}|\pi_{ij})$ and $p(x_{ijk}|\epsilon)$ is joint computed by belief update. The learning rate $\gamma = (1+n)^{-\alpha}$ controls the weighting of new cases and it is determined by n and α. [2] suggests to use $\alpha = 0.6$ while [18] recommends $\alpha = 0.501$.

Online EM, which can be considered a gradient ascent algorithm, was described by [2].

The fractional updating algorithm, see e.g., [16] who cites[21], Fading of past cases can be controlled using a fading factor λ specified for each parent configuration.

In the general case of OOBNs, we compute the average expected count for the run-time instances of the node and increase the experience count by the number of run-time instances.

6 Empirical Evaluation

In this section, we describe the results of an empirical evaluation of the feasibility of using parameter learning algorithms to improve the diagnostic performance on the Linear Axis based on operational data.

6.1 Experimental Setup

The empirical evaluation is organised into three different tests (1) we consider updating the parameters of the knowledge driven model using the real operational dataset, (2) we consider updating the parameters of the knowledge driven model where all distributions are made uniform using a dataset of $250,000$ cases with 5% missing values generated completely at random from the knowledge driven model, and (3) we consider the time performance of updating the parameters in the knowledge driven model. The evaluations are performed using different values of the parameters of the learning algorithms.

6.2 Experimental Results

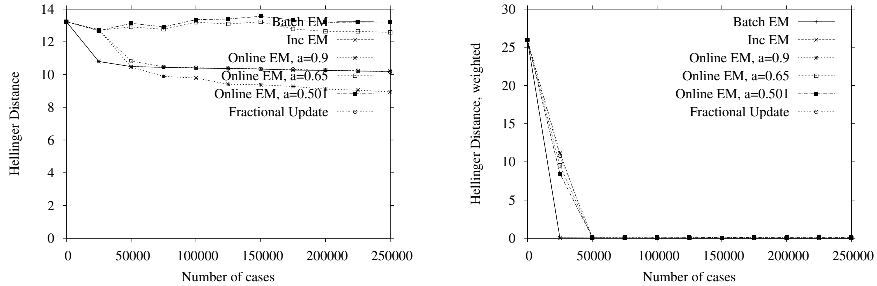
The operational dataset is rather sparse. The dataset contains 13429 cases with six observed sensor readings represented in the model. The dataset contains failure and non-failure cases.

Table 1 shows the diagnostic performance of the five models considered where μ_{rank} refers to the average rank of the *true* root cause, i.e., the value 1 means perfect performance and 28 worst possible performance.

Table 1. The diagnostic performance of the five models considered.

Algorithm	Top-1	Top-5	μ_{rank}
Knowledge driven model	8	19	5
Batch EM			
Incremental EM			
Online EM			
Fractional update			

One root cause has *zero* probability in the model - Moter bearings - how do we generate data them???? take it out and run again



(a) Operational Data Sample (b) Operational Data Sample

Fig. 4. Bla bla.

Next, we report on an experimental analysis of the performance of different levels of integration of the IEF model into the SelSus architecture. The tightest level of integration has been achieved by integrating the model directly into the component control software. In addition, the model has been deployed using a web service inside the SelSus Cloud having the control software and web service running on the same machine as well as having the control software and web service running on machines located far apart (more than 1000km).

Table 2. Average time cost of one belief update and learning cycle across the integration levels.

Algorithm	Configuration	Total time (ms)	Average time (ms)
Online EM	Direct integration	1,508	0.189
	Direct integration (w/save-to-memory)	778	0.097
	Localhost deployment	10,263	1.283
	Network deployment	382,785	47.848
Fractional Updating	Direct integration	1,508	0.189
	Direct integration (w/save-to-memory)	778	0.097
	Localhost deployment	10,263	1.283
	Network deployment	382,785	47.848

Table 2 *In the experiment, one state for each possible observation was propagated and this process was repeated* 1000 *times producing* 8000 *propagations. Only Online EM and fractional updating ???*

7 Discussion

here we discuss the insights and make recommendations as well as relate our results to [18]

Here are some ideas for topics to discuss:

- relate to [18] results
- data recommendations
- which conclusions on the data
- (too little failure)
- what should we conclude
- bias - no failures
- Data annotation (WP3), data model (T6.1) that implement data parser
- data preprocessing component level. DAta preprocessing on the component level before feeding into hugin
- weighting data compare to the elicitation from expert
- instead of online learning, it can be performed under supervision of the operator / expert to build confidence

Acknowledgments

This work is part of the project "Health Monitoring and Life-Long Capability Management for SELf-SUStaining Manufacturing Systems (SelSus)" which is funded by the Commission of the European Communities under the 7th Framework Programme, Grant agreement no: 609382. We thank IEF-Werner GmbH for allowing us to use the Linear Axis model and data.

References

1. J. Binder, D. Koller, S. Russell, and K. Kanazawa. Adaptive probabilistic networks with hidden variables. *Machine Learning*, 29(2):213–244, 1997.
2. O. Cappe and E. Moulines. Online em algorithm for latent data models. *Journal of the Royal Statistical Society Series B (Statistical Methodology)*, 71(3):593–613, 2009.
3. R.G. Cowell, A.P. Dawid, S.L. Lauritzen, and D.J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer, 1999.
4. F. V. Jensen. Gradient Descent Training of Bayesian Networks. In *Proc. of the 5th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 190–200, 1999.
5. U. B. Kjærulff and A. L. Madsen. *Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis*. Springer, 2nd edition, 2013.
6. G. Kokolakis and P. Nanopoulos. Bayesian multivariate micro-aggregation under the hellingers distance criterion. *Research in Official Statistics*, 4(1):117–126, 2001.
7. D. Koller and A. Pfeffer. Object-oriented bayesian networks. In *Proceedings of the Thirteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-97)*, pages 302–313, 1997.
8. S. L. Lauritzen. The EM algorithm for graphical association models with missing data. *Computational Statistics & Analysis*, 19:191–201, 1995.
9. A. L. Madsen, N. Søndberg-Jeppesen, N. Lohse, and M. Sayed. A methodology for developing local smart diagnostic models using expert knowledge. In *INDIN 2015 IEEE International Conference on Industrial Informatics*, pages 1682–1687, 2015.
10. A. L. Madsen, N. Søndberg-Jeppesen, N. Lohse, and M. S. Sayed. A methodology for developing local smart diagnostic models using expert knowledge˙In *Proceedings of the 2015 IEEE International Conference on Industrial Informat ics (INDIN)*, page 6, 2015.

11. A. L. Madsen, N. Søndberg-Jeppesen, M. S. Sayed, M. Peschl, and N. Lohse. Applying object-oriented bayesian networks for smart diagnosis and health monitoring at both component and factory level. In *Under Review*, page 10, 2017.
12. A.L. Madsen, M. Lang, U.B. Kjærulff, and F. Jensen. *The Hugin Tool for Learning Bayesian Networks*, pages 594–605. 2003.
13. M. Neil, N. Fenton, and L. M. Nielsen. Building large-scale Bayesian networks. *The Knowledge Engineering Review*, 15(3):257–284, 2000.
14. L. Neto, J. Reis, D. Guimaraes, and G. Concalves. Sensor cloud: Smartcomponent framework for reconfigurable diagnostics in intelligent manufacturing environments. In *Proceedings of the 2015 IEEE International Conference on Industrial Informat ics (INDIN)*, pages 1706–1711, 2015.
15. L. Neto, J. Reis, R. Silva, and G. Concalves. Sensor selcomp, a smart component for the industrial sensor cloud of the future. In *Under Review*, page 6, 2017.
16. K. G. Olesen, S. L. Lauritzen, and F. V. Jensen. aHUGIN: A System Creating Adaptive Causal Probabilistic Networks. In *Proc. of the 8th Conference on UAI*, pages 223–229, 1992.
17. J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Series in Representation and Reasoning. Morgan Kaufmann Publishers, San Mateo, CA, 1988.
18. P. Ratnapinda and M.J. druzdzel. Learning discrete bayesian network parameters from continuous data streams: What is the best strategy? *Journal of Applied Logic*, 13:628–642, 2015.
19. S. Russell, J. Binder, D. Koller, and K. Kanazawa. Local learning in probabilistic networks with hidden variables. In *"Proc. of the Thirteenth International Joint Conference on Artificial Intelligence"*, pages 1146–1152, 1995.
20. M. S. Sayed, N. Lohse, N. Søndberg-Jeppesen, and A. L. Madsen. Selsus: Towards a reference architecture for diagnostics and predictive maintena nce using smart manufacturing devices. In *Proceedings of the 2015 IEEE International Conference on Industrial Informati cs (INDIN)*, page 6, 2015.
21. D.M. Titterington. Updating a diagnostic system using unconfirmed cases. *Applied Statistics*, 25:238–47, 1976.
22. A. Zagorecki, M. Voortman, and M.J. Druzdzel. Decomposing local probability distributions in bayesian networks for improved inference and parameter learning. In *Proceedings of the 2006 FLAIRS conference*, pages 860–865, 2006.